What is a proof?

What is a proof?

By Alan Bundy[1] and Mateja Jamnik[2] and Andrew Fugard[1]

[1]*School of Informatics, University of Edinburgh, Appleton Tower, Crichton St, Edinburgh EH8 9LE, UK*

[2]*University of Cambridge Computer Laboratory, J.J. Thomson Avenue, Cambridge CB3 0FD, UK*

To those brought up in a logic-based tradition there seems to be a simple and clear definition of proof. But this is largely a 20^{th} century invention; many earlier proofs had a different nature. We will look particularly at the faulty proof of Euler's Theorem and Lakatos' rational reconstruction of the history of this proof. We will ask: how is it possible for the errors in a faulty proof to remain undetected for several years – even when counter-examples to it are known? How is it possible to have a proof about concepts that are only partially defined? And can we give a logic-based account of such phenomena? We introduce the concept of *schematic proofs* and argue that they offer a possible cognitive model for the human construction of proofs in mathematics. In particular, we show how they can account for persistent errors in proofs.

Keywords: mathematical proof, automated theorem proving, schematic proof, constructive omega rule.

1. Introduction

To those brought up in a logic-based tradition there seems to be a simple and clear definition of proof. Paraphrasing Hilbert (Hilbert 1930):

> *A proof is a sequence of formulae each of which is either an axiom or follows from earlier formulae by a rule of inference.*

Let us call a proof in this format *Hilbertian*.

But formal logic and its Hilbertian view of proof is largely a 20^{th} century invention. It was invented to help avoid erroneous proofs and to enable proofs about proofs, for instance Gödel's proof of the incompleteness of arithmetic (Gödel 1931). Formal logic has since become the basis for automated theorem proving.

Prior to the invention of formal logic, a proof was any convincing argument. Indeed, it still is. Presenting proofs in Hilbertian style has never taken off within the mathematical community. Instead, mathematicians write *rigorous* proofs, i.e. proofs in whose soundness the mathematical community has confidence, but which are not Hilbertian.

To see that rigorous proofs are not Hilbertian, consider erroneous proofs. The history of mathematics is full of erroneous proofs. The faults in some of these proofs have remained undetected or uncorrected for many years. However, Hilbertian proofs can be readily checked. We merely need to ask of each formula in the

proof: 'is it an axiom?' or 'does it follow from earlier formulas by a rule of inference?'. Such checking can be readily automated – and, indeed, it often has been. But even in the age before computers, postgraduate students could have carried out the necessary checks as an apprentice exercise. Mathematical proofs are subjected to a lot of checking, so we can conclude that proofs containing persistent errors are unlikely to be Hilbertian. If a Hilbertian proof contained an error, it would surely be quickly detected and corrected.

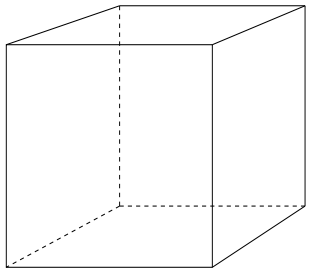
This may lead us to ask, what are the alternatives to Hilbertian proof presentation? Can these alternative presentations help us to understand how a fault can lie undetected or uncorrected for a long time? Could we formalise these alternative presentations? Could we automate such mathematical reasoning?

We will present an alternative method of proof presentation that we call *schematic proof*. Schematic proof is partly inspired by Lakatos's rational reconstruction of the history of Euler's Theorem (Lakatos 1976). It has been automated in two projects at Edinburgh, using the constructive ω-rule (Baker 1993, Jamnik 2001).

We argue that schematic proofs offer a possible cognitive model for the human construction of proofs in mathematics. In particular, we show how they can account for persistent errors in proofs. They also help explain a key role for examples in the construction of proofs and a paradox in the relative obviousness of different, but related, theorems.

2. Lakatos's discussion of Euler's theorem

Euler's Theorem† states that, in any polyhedron, $V - E + F = 2$, where V is the number of vertexes, E is the number of edges and F is the number of faces. This theorem is illustrated in the case of the cube in figure 1.



In a cube there are 8 vertexes, 12 edges and 6 faces. So $V - E + F = 8 - 12 + 6 = 2$.

Figure 1. Euler's Theorem in the case of the cube

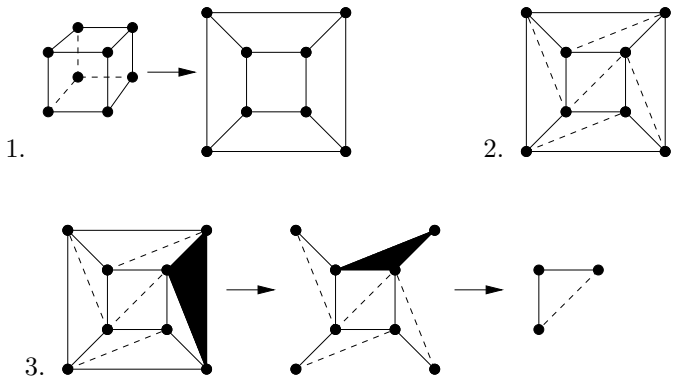
In (Lakatos 1976), Imre Lakatos gives a rational reconstruction of the history of Euler's Theorem. This history is set in a fictitious classroom of extremely bright students and a teacher. The students adopt different roles in the history of the evolution of mathematical methodology. The teacher leads them through this history.

† More properly, this would be called 'Euler's Conjecture', since he proposed, but did not prove it.

To initiate the discussion, the teacher presents a 'proof' of Euler's Theorem due to Cauchy, which we reproduce in figure 2. Later, the teacher and the students discover various counter-examples to the Theorem. They use these counter-examples to analyse the faulty proof and propose a wide variety of different methods for dealing with the conflict between alleged proofs and counter-examples. These methods involve refining definitions, correcting proofs, modifying conjectures, etc. Lakatos's mission was to show how the methodology of mathematics had evolved: becoming more sophisticated in its handling of proofs and refutations. Our mission is different: we will analyse the proof method underlying Cauchy's faulty proof and show how errors can arise from the use of this proof method.

(*a*) *Cauchy's 'proof' of Euler's theorem*

Lakatos's account of Cauchy's 'proof' is illustrated in figure 2 for the case of the cube. The general 'proof' is given in Theorem 2.1.



In step 1, one face of the cube is removed and the remaining faces are stretched onto the plane. In step 2, these faces are triangulated to break them into triangles. In step 3, these triangles are removed one by one. Two cases of step 3 can arise and are illustrated. The figures are adapted from (Lakatos 1976).

Figure 2. Cauchy's 'proof' applied to the cube

Theorem 2.1. *For any polyhedron, $V - E + F = 2$, where V is the number of vertexes, E is the number of edges and F is the number of faces.*

Cauchy's 'proof'. Given a polyhedron, carry out the following steps.

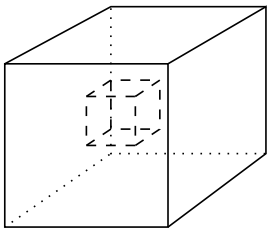
1. *Remove one face and stretch the other faces onto the plane.* Note that F has diminished by 1, but that V and E are unchanged. So we are required to prove that $V - E + F = 1$.
2. *Triangulate the remaining faces by drawing diagonals.* Note that each new diagonal increases both E and F by 1, but leaves V unchanged. So $V - E + F$ is unaffected.

3. *Remove the triangles one by one.* There are two cases to consider, illustrated by step 3 in figure 2. In the first case, we remove an edge, so that both E and F decrease by 1. In the second case, we remove two edges and a vertex, so that both V and F decrease by 1, but E decreases by 2. In either case, $V - E + F$ is unaffected.

Finally, we are left with one triangle. In a triangle, $V = 3$, $E = 3$ and $F = 1$, so $V - E + F = 1$, as required. □

(*b*) *A counter-example to Euler's theorem*

Many people found Cauchy's 'proof' convincing. However, as Lakatos illustrates, eventually many counter-examples were found. The simplest is the hollow cube, illustrated in figure 3.



The hollow cube is a cube with a cubical hole in the middle. The values of V, E and F are all doubled. So $V - E + F = 16 - 24 + 12 = 4$.

Figure 3. The hollow cube: a counter-example to Cauchy's 'proof'

Lakatos reports the reaction to this counter-example as a debate about whether the hollow cube is really a polyhedron. He offers two possible definitions of *polyhedron*, providing two opposite answers to this question.

Definition 1: *A polyhedron is a* solid *whose surface consists of polygonal faces.* Under this definition, the hollow cube *is* a polyhedron.

Definition 2: *A polyhedron is a* surface *consisting of a system of polygons.* Under this definition, the hollow cube *is not* a polyhedron.

It is interesting to ask how it is *possible* for Cauchy to have offered a proof about polyhedra, when the question of their definition was still open.† This could not happen in Hilbertian proofs: definitions are axioms and must come first. What kind of proof *might* allow us to keep the definitions open?

3. Schematic proofs

Note that the proof of theorem 2.1 is a *procedure*: given a polyhedron, a series of operations is specified, whose application will reduce the polyhedron to the triangle.

† Nor is it yet closed, since terms such as *surface*, *system*, etc., have still to be defined.

The value of $V - E + F$ is tracked during these operations. The actual number of operations to be applied will vary depending on the input polyhedron. This is very *unlike* a Hilbertian proof, which is not procedural and in which the same number of proof steps is used for all examples.

Independently of (Lakatos 1976), the Mathematical Reasoning Group at Edinburgh became interested in the constructive ω-rule. We subsequently realised that this rule generates just the kind of proof used above to prove theorem 2.1. We call this kind of proof *schematic*. Schematic proofs are procedures that, given an example, generate a proof which is specific to that example. The number of steps in each proof depends on the example.

(*a*) *The constructive ω-rule*

The ω-rule for the natural numbers 0, 1, 2, ... is:

$$\frac{\phi(0), \quad \phi(1), \quad \phi(2), \quad \ldots}{\forall x.\phi(x)}$$

i.e. we can infer that $\phi(x)$ for all natural numbers x provided we can prove $\phi(n)$ for $n = 0, 1, 2, \ldots$. The ω-rule is clearly not a very practical rule of inference, since it requires the proof of an infinite number of premises to prove its conclusion. A Hilbertian proof using it would consist of an infinite sequence of formulas. Its use is usually confined to theoretical discussions, for instance, in Gödel's incompleteness theorems (Gödel 1931).

The constructive ω-rule is a refinement of the ω-rule that *can* be used in practical proofs. It has the additional requirement that the $\phi(n)$ premises be proved in a *uniform* way, i.e. that there exists a recursive program, proof_ϕ, which takes a natural number n as input and returns a proof of $\phi(n)$ as output. We will write this as $\mathit{proof}_\phi(n) : \phi(n)$. The recursive program proof_ϕ formalises our notion of schematic proof. Applied to the domain of polyhedra, rather than natural numbers, it could be used to formalise Cauchy's 'proof' of Euler's theorem given in theorem 2.1 in §2(a).

(*b*) *Implementation of the constructive ω-rule*

To demonstrate its practicality as a rule of inference, we have implemented the constructive ω-rule within two automated theorem-proving systems. In outline, these automated theorem-provers use the following procedure.

1. Start with some proofs for specific examples, e.g. $\mathit{proof}_\phi(3){:}\phi(3)$, $\mathit{proof}_\phi(4){:}\ \phi(4)$.

2. Generalise these proofs of examples to obtain a recursive program: proof_ϕ.

3. Verify, by induction, that this program constructs a proof for each n.

$$\mathit{proof}_\phi(0) : \phi(0)$$
$$\mathit{proof}_\phi(n) : \phi(n) \vdash \mathit{proof}_\phi(n+1) : \phi(n+1)$$

$$
\begin{array}{rcl}
(s^m(0) + s^n(0)) + s^r(0) & = & s^m(0) + (s^n(0) + s^r(0)) \\
+_s \times m \quad \vdots & & +_s \times m \quad \vdots \\
s^m(0 + s^n(0)) + s^r(0) & = & s^m(0 + (s^n(0) + s^r(0))) \\
+_b \quad \downarrow & & +_b \quad \downarrow \\
s^m(s^n(0)) + s^r(0) & = & s^m(s^n(0) + s^r(0)) \\
+_s \times m \quad \vdots & & \\
s^m(s^n(0) + s^r(0)) & = & s^m(s^n(0) + s^r(0))
\end{array}
$$

$s(n)$ is the successor function for natural numbers, intuitively meaning $n+1$. $s^m(0)$ means s applied m times to 0. Addition is defined recursively using two equations: the base case $0+y = y$ and the step case $s(x)+y = s(x+y)$. $+_s$ is rewriting, left to right, using this step case; $+_b$ is rewriting using the base case. Note that the number of applications of $+_s$ depends on m.

Figure 4. Schematic proof of the associativity of +

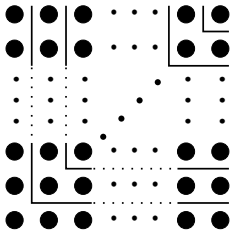
At first sight it may appear that step 3 replaces a Hilbertian, object-level induction with an isomorphic meta-level induction. Siani Baker's work (Baker 1993), however, shows that the meta-level induction is *not* isomorphic to the object-level one; often it is much simpler. Her work revealed many examples in which the object-level proof required generalisation or intermediate lemmas, but the meta-level proof did not, i.e. the meta-level proof was inherently simpler.

Note that Cauchy's 'proof' omits the final verification step 3 in the above procedure. He is trusting that his program for reducing polyhedra to triangles will work for all polyhedra. As we have seen, it doesn't. This helps explain the error in his proof.

Both of our implementations of the constructive ω-rule were for the natural numbers: Siani Baker's for simple arithmetic theorems (Baker 1993) and the second author's for a form of diagrammatic reasoning (Jamnik 2001), as described in Roger Nelsen's book 'Proofs without words' (Nelsen 1993). Both our implementations included the final verification step, so were guaranteed to produce only correct schematic proofs.

An example schematic proof from Baker's work is given in figure 4. Rather than describe the program in a programming language, we try to capture the infinite family of proofs that it outputs, using ellipses to indicate those steps and expressions that occur a variable number of times.

An example proof from the second author's work is shown in figure 5. Her program, Diamond, was shown proofs for two numbers and then generalized these into a program for generating the proof for any number.



The diagram gives a proof of the theorem $n^2 = 1+3+\cdots+(2n-1)$. *The diagram can be viewed as describing both the left and right hand sides of the equation. The whole square represents* n^2. *Each of the L-shapes represents one of the odd numbers summed on the right-hand side.*

Figure 5. A proof without words

4. The relative difficulty of proofs

Proof by induction is the Hilbertian alternative to schematic proofs of theorems about recursive data-types, such as the natural numbers. We have also investigated the automation of inductive proofs (Bundy 2001). However, evidence arising from these investigations suggests that humans do *not* use inductive proof when assessing the truth or falsity of conjectures. This is perhaps not surprising, since the formalisation of mathematical induction is a relatively modern development†. *Schematic proof* is an alternative candidate model of the mechanism humans use to prove conjectures over infinite domains.

To illustrate the evidence against inductive proof as a cognitive model, consider the *rotate-length* theorem on lists:

$$\forall l \in list(\tau).\ rot(len(l), l) = l, \tag{4.1}$$

where l is a list of elements of type τ, len is a unary function that takes a list l and returns its length, and rot is a binary function that takes a number n and list l and rotates the first n elements of l from the front to the back. The recursive definitions of len and rot are given in figure 6. Also defined, is a binary function $<>$, which takes two lists and appends them together. $<>$ is an auxiliary function in the definition of rot. The most straightforward induction rule for lists is:

$$\frac{\Phi([\,]) \quad \forall h \in \tau. \forall t \in list(\tau).\ \Phi(t) \rightarrow \Phi([h|t])}{\forall l \in list(\tau).\ \Phi(l)}$$

where $[\,]$ is the empty list, $[h|t]$ places an h at the front of a list t, τ is an arbitrary type and $list(\tau)$ is the type of lists of elements of type τ.

Having absorbed the definitions, most people will readily agree that the rotate-length theorem 4.1 is true. However, its inductive proof is surprisingly difficult. It cannot be proved directly from the recursive definitions in figure 6. The proof requires the use of auxiliary lemmas, the generalization of the theorem, and/or

† It is usually attributed to Richard Dedekind in 1887, although informal uses of induction date back as far as Euclid.

$$
\begin{aligned}
[\,] <> K &= K \\
[H|T] <> K &= [H|T <> K]
\end{aligned}
$$

$$
\begin{aligned}
len([\,]) &= 0 \\
len([H|T]) &= s(len(T))
\end{aligned}
$$

$$
\begin{aligned}
rot(0, L) &= L \\
rot(s(N), [\,]) &= [\,] \\
rot(s(N), [H|T]) &= rot(N, T <> [H])
\end{aligned}
$$

Each definition consists of one or more base and step cases. The base cases define the function for one or more initial values. The step cases define the function for constructed values in terms of the parts from which they are constructed.

Figure 6. The recursive definitions of some functions

the use of a more elaborate form of induction. For instance, one way to prove the rotate-length theorem is first to generalise it to:

$$\forall k \in list(\tau). \forall l \in list(\tau).\ rot(len(l), l <> k) = k <> l. \tag{4.2}$$

Although people will also readily agree that (4.2) is also true, they find this assessment a little more difficult that that of (4.1).

So, if people are using induction to assess conjectures such as (4.1) and (4.2), even if unconsciously, then we are faced with a paradox: what appears to be a fairly easy assessment of (4.1), entails what appears to be the harder assessment of (4.2). Moreover, the inductive proof is quite difficult, requiring, for instance, the generalization of the initial conjecture and the speculation and proof of a couple of intermediate lemmas (or some alternative but similarly complex processes) (Bundy 2001, §6.3.3, §6.2.2). This phenomenon is not rare. On the contrary, we have found lots of similar examples, where an intuitively obvious conjecture has only a complex inductive proof, requiring generalization, lemmas, non-standard inductions, or a mixture of these. The intermediate generalizations, lemmas, etc., are often harder to assess than the original conjecture.

The schematic proof of the rotate-length theorem is given in figure 7. This does *not* require any generalizations or intermediate lemmas, and is fairly straightforward. The schematic proof can be viewed as evaluating the theorem on a generic list using the recursive definitions. In informal experiments, when we have asked subjects what mechanism they used to assess the truth/falsity of this theorem, they report a process that resembles this schematic proof,† making it a candidate for a cognitive model.

† For instance, Aaron Sloman, personal communication.

$$
\begin{array}{rcl}
rot(len([a_1, a_2, \ldots, a_n]), [a_1, a_2, \ldots, a_n]) & = & [a_1, a_2, \ldots, a_n] \\
(len_s \times n) + len_b \quad \downarrow & & \\
rot(s^n(0), [a_1, a_2, \ldots, a_n]) & = & [a_1, a_2, \ldots, a_n] \\
rot_s \quad \downarrow & & \\
rot(s^{n-1}(0), [a_2, \ldots, a_n] <> [a_1]) & = & [a_1, a_2, \ldots, a_n] \\
(<>_s \times (n-2)) + <>_b \quad \vdots & & \\
rot(s^{n-1}(0), [a_2, \ldots, a_n, a_1]) & = & [a_1, a_2, \ldots, a_n] \\
repeat \times (n-1) \quad \vdots & & \\
rot(0, [a_1, a_2, \ldots, a_n]) & = & [a_1, a_2, \ldots, a_n] \\
rot_b \quad \downarrow & & \\
[a_1, a_2, \ldots, a_n] & = & [a_1, a_2, \ldots, a_n]
\end{array}
$$

$<>_s$, len_s and rot_s refer to rewriting using the step cases of the definitions of these functions. Similarly, $<>_b$, len_b and rot_b refer to rewriting using the base cases. Note that the number of times these rules are applied depends on n, the length of the list. However, only rewriting with the recursive definitions is required.

Figure 7. A schematic proof of the rotate-length theorem

5. Schematic proofs as a cognitive model

We hypothesise that schematic proofs provide a cognitive model of some mathematical proofs. We are soon to start a project to test this hypothesis. It appears to provide a good fit to procedural-style proofs, such as Cauchy's 'proof' of Euler's Theorem and some informal justifications of inductive conjectures. It gives an account of how certain kinds of errors might occur: if the final verification step is omitted, then it is a matter of luck whether a schematic proof will produce a sound proof for every example. If the number of possible examples is infinite or very large, it may take some considerable time before someone comes across an example for which the schematic proof fails. It may be difficult to tell *why* the schematic proof fails on this example and how to repair it.

The constructive ω-rule provides a logic-based, formal account of schematic proof, as an alternative to the standard Hilbertian account. This gives us a sound mathematical basis to investigate rigorous†, as opposed to Hilbertian, proof. It fills a gap in MacKenzie's classification of proofs (this volume) by providing an example of a rigorous but mechanisable style of proof.

To confirm our hypothesis, we will have to carry out a series of psychological experiments to compare human mathematical performance with our implementations and formal accounts of schematic proof. Here is an outline of the kind of experiment we might perform.

† Rigorous proofs are sometimes called "informal", but this is a misleading description since, as we have seen, they can be formalised.

- Present theorems and near-miss non-theorems to human mathematicians.
- Ask for both their judgement of the truth/falsity of each example, together with a justification of that decision.
- Investigate how subjects explore and absorb the meaning of recursive definitions. Do they try examples instances? Do they reason inductively? We may provide a computational tool that will support subject's exploration and keep a record of these explorations as experimental data.
- Try to model these justifications and explorations computationally.
- Decide whether schematic proof or a Hilbertian proof provides a better explanation of these justifications, or whether neither is a good fit and there is a third possibility.† It may be that subjects' preferences vary according to their reasoning style. We may want to apply some pre-tests to classify the subjects' reasoning styles.

There has been some previous work on building computational models for how students recognise patterns in numbers, and construct functions to generate the numbers (Haverty *et al* 2000). This work could provide guidance on our experimental design, for instance, asking the students to speak aloud during the experiment, then analysing the resulting protocols for clues about internal processing.

We also plan to conduct an investigation of some historical proofs, especially those that were later found to be faulty, to see if schematic proof can offer an explanation of the mechanism used, and help explain why errors were made that were hard to detect or correct.

6. Discussion

In this section we discuss some related work.

(*a*) *Comparison to Type Theory*

There are superficial similarities between the proposals here, to use the constructive ω-rule to produce schematic proofs, and the formal representation of proofs in constructive type theory (Martin-Löf 1984). Both, for instance, associate, with each theorem, an object that can be interpreted as both a program and a proof. However, there are also significant differences.

Firstly, the object associated with a theorem in type theory can be interpreted both as a program constructed by the proof and as a verification proof of the correctness of that program and of the theorem. Moreover, the type theory proof proves the *whole* theorem. On the other hand, the program associated with a theorem by the constructive ω-rule is not a general proof of the theorem; it is a program which will *generate* a putative proof for each instance of the theorem. Note that members of this family of proofs can contain different numbers of steps.

Secondly, the program in type theory is generated by the process of proving the theorem. Its correctness is guaranteed by the soundness of type theory. The

† Another potential candidate is a mechanism we have investigated for reasoning with ellipsis (Bundy & Richardson 1999).

program in the constructive ω-rule is generated by inductive generalisation from a few example proofs of these theorem instances. As we have shown, its correctness is *not* guaranteed: an additional meta-level proof is required to establish this. This meta-level proof has no counterpart in type theory.

(*b*) *Rigorous Proof as Hilbertian Proof Highlights*

Many accounts of rigorous proof implicitly or explicitly adopt the position that a rigorous proof is essentially a Hilbertian proof, but with steps missing: possibly, 90% or more of the steps. This is probably a valid account of many human-constructed proofs. We must then ask how errors may arise in such proofs, and how these errors may lie undetected.

It is not the case that mathematicians first produce the Hilbertian proofs and then summarise them for publication by eliding 90%+ of the steps. Firstly, this explanation is at variance with accounts of proof discovery. Secondly, the few attempts to turn rigorous proofs into Hilbertian proofs often reveal errors. For instance, Jacques Fleuriot's formalisation of Newton's proofs of Kepler's Laws, using the Isabelle prover and Non-Standard Analysis (Fleuriot 2001), revealed an error in Newton's manipulation of infinitesimal numbers. Even Hilbert himself was not immune. Laura Meikle's Isabelle formalisation of Hilbert's Grundlagen, revealed that Hilbert had appealed to the semantics of the geometry domain rather than just the axioms and rules of the formal theory. Thirdly, rigorous proofs often prove unreasonably hard to check. I have argued that checking Hilbertian proofs should be routine: even Hilbertian proofs with many missing steps. At the time of writing, putative proofs of both the Poincaré Conjecture and the Kepler Conjecture are undergoing extensive checking.

Human accounts of proof discovery suggest that mathematicians first form a plan, which they then unpack until they are satisfied of the truth of each proof step. This process of proof discovery can also be automated using the technique of *proof planning* (Bundy 1991). However, in our automation of proof planning, the plan is unpacked into a Hilbertian proof. Humans stop short of this level of unpacking. It would be interesting to investigate how they decide when to stop. A possibility worth investigation is that schematic proof is used at the leaves of the proof plan, i.e. that the proof plan is unpacked until the remaining subgoals can be checked against a few well-chosen examples. This would explain how errors could be introduced into the proof plan. It also unites our two rival accounts of rigorous proof.

7. Conclusion

The standard Hilbertian account of mathematical proof fails to model some historically important proofs, to account for the possibility of undetected and uncorrected error and to account for the relative difficulty of proofs. Schematic proof provides an alternative account of proof that *does* address these issues. Schematic proofs are based on the constructive ω-rule, which provides a formal, logic-based foundation. This, for instance, enables us to automate the construction and application of schematic proofs. Schematic proof provides a link to computer program verification, in which an *invariant* formula, analogous to $V - E + F$, is shown to be preserved

by successive computational operations. Just like Cauchy, programmers who do not verify their programs run the risk that their systems will fail on unforeseen inputs. The process of forming schematic proofs by generalising from examples provides a key role for examples in the construction of proofs. This may go some way to explain why humans find models, diagrams, *etc* so valuable during proof discovery.

We are now planning to conduct some psychological investigations into the extent to which schematic proofs can account for the mechanisms of human proof discovery.

Acknowledgements

The research reported in this paper was supported by EPSRC grants GR/S01771 and GR/S31099 (Bundy), an EPSRC Advanced Research Fellowship GR/R76783 (Jamnik), and a Swedish Institute Guest Scholarship and EPSRC/MRC Neuroinformatics Studentship EP/C51291X/1 (Fugard). We are grateful to Jürgen Zimmer for help with (Hilbert 1930).

References

Baker, S. (1993), *Aspects of the Constructive Omega Rule within Automated Deduction*, PhD thesis, University of Edinburgh, Edinburgh, UK.

Bundy, A. (1991)A science of reasoning. In Lassez, J.-L. and Plotkin, G., (eds.), *Computational Logic: Essays in Honor of Alan Robinson*, pages 178–198. MIT Press.

Bundy, A. (2001), The automation of proof by mathematical induction, *in* A. Robinson & A. Voronkov, eds, 'Handbook of Automated Reasoning, Volume 1', Elsevier.

Bundy, A. & Richardson, J. (1999), Proofs about lists using ellipsis, *in* H. Ganzinger, D. McAllester & A. Voronkov, eds, 'Proceedings of the 6th International Conference on Logic for Programming and Automated Reasoning, LPAR', number 1705 *in* 'Lecture Notes in Artificial Intelligence', Springer Verlag, pp. 1–12.

Fleuriot, J. (2001)*A Combination of Geometry Theorem Proving and Nonstandard Analysis, with Application to Newton's Principia*. Springer-Verlag.

Gödel, K. (1931), 'Über formal unentscheidbare sätze der Principia Mathematica und verwandter systeme i', *Monatsh. Math. Phys.* **38**, 173–198. English translation in (Heijenoort 1967).

Heijenoort, J. V. (1967), *From Frege to Gödel: a source book in Mathematical Logic, 1879-1931*, Harvard University Press, Harvard, MA.

Haverty, L.A., Koedinger, K.R., Klahr, D. and Alibali, M.W. (2000)Solving inductive reasoning problems in mathematics: not-so-trivial pursuit. *Cognitive Science*, 24(2):249–298.

Hilbert, D. (1930), 'Die Grundlebung der elementahren Zahlenlehre', *Mathematische Annalen*, **104**, 485-494.

Jamnik, M. (2001), *Mathematical Reasoning with Diagrams: From Intuition to Automation*, CSLI Press, Stanford, CA.

Lakatos, I. (1976), *Proofs and Refutations: The Logic of Mathematical Discovery*, Cambridge University Press, Cambridge, UK.

Martin-Löf, Per. (1984)*Intuitionistic Type Theory*. Bibliopolis, Naples, Notes by Giovanni Sambin of a series of lectures given in Padua, June 1980.

Meikle, L. I. and Fleuriot, J. D. (2003)Formalizing Hilbert's Grundlagen in Isabelle/Isar. In *Theorem Proving in Higher Order Logics: 16th International Conference, TPHOLs 2003*, volume 2758 of *Lecture Notes in Computer Science*, pages 319–334.

Nelsen, R. (1993), *Proofs without Words: Exercises in Visual Thinking*, Mathematical Association of America, Washington, DC.